ADST-II-CDRL-MC130-2000071
April 18, 2000

ADVANCED DISTRIBUTED SIMULATION TECHNOLOGY II (ADST II)

High Level Architecture (HLA) Implementation For the MC-130E and MC-130H Combat Talons DO #105

CDRL AB02 FINAL REPORT AND IMPLEMENTATION RESULTS

For:

United States Army
Simulation, Training, and Instrumentation Command
12350 Research Parkway
Orlando, Florida 32826-3224
Attn: AMSTI-ED

By:

Science Applications International Corporation
12479 Research Parkway
Orlando, FL 32826-3248

Lockheed Martin
Information Systems Company
12506 Lake Underhill Road
Orlando, FL 32825

UNCLASSIFIED

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 074-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	18 APR 2000	FINAL

4. TITLE AND SUBTITLE
Advanced Distributed Simulation Technology II (ADST-II) High Level Architecture Implementation for the MC-130E and MC-130H combat Talons Final Report & Implementation Results

5. FUNDING NUMBERS
N61339-96-D-0002

6. AUTHOR(S)

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Lockheed Martin Information Systems
ADST-II
P.O. Box 780217
Orlando Fl 32878-0217

8. PERFORMING ORGANIZATION REPORT NUMBER
ADST-II-CDRL-MC130-2000071

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

NAWCTSD/STRICOM
12350 Research Parkway
Orlando, FL 32328-3224

10. SPONSORING / MONITORING AGENCY REPORT NUMBER
CDRL AB02

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for Public release; distribution is unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT ***(Maximum 200 Words)***
The High Level Architecture Applications Implementation for MC-130E and MC-130H Combat Talon simulators project was executed as Delivery Order 105 by the Advanced Distributed Simulation Technology II (ADST II) integrated product team. The primary objective was to advance the interoperability of the Special Operations Forces (SOF) training and mission rehearsal simulators. The project incorporated the DoD High Level Architecture (HLA) into the MC-130E and MC-130H Combat Talon simulators at the 19th Special Operations Squadron (SOS) at Hurlburt Field, FL. It was sponsored by the U.S. Special Operations Command (USSOCOM) and administered by the U.S. Army Simulation, Training and Instrumentation Command (STRICOM).

14. SUBJECT TERMS
STRICOM, ADST-II, MC-130E

15. NUMBER OF PAGES
54

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT UNCLASSIFIED	18. SECURITY CLASSIFICATION OF THIS PAGE UNCLASSIFIED	19. SECURITY CLASSIFICATION OF ABSTRACT UNCLASSIFIED	20. LIMITATION OF ABSTRACT

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. Z39-18
298-102

DTIC QUALITY INSPECTED 4

Document Control Information

Revision	Revision History	Date
	Original release	4/21/2000
-		
-		
-		
-		
-		
-		
-		
-		
-		
-		
-		
-		
-		

Table Of Contents

List of Figures

EXECUTIVE SUMMARY

The High Level Architecture Applications Implementation for MC-130E and MC-130H Combat Talon simulators project was executed as Delivery Order 105 by the Advanced Distributed Simulation Technology II (ADST II) integrated product team. The primary objective was to advance the interoperability of the Special Operations Forces (SOF) training and mission rehearsal simulators. The project incorporated the DoD High Level Architecture (HLA) into the MC-130E and MC-130H Combat Talon simulators at the 19th Special Operations Squadron (SOS) at Hurlburt Field, FL. It was sponsored by the U.S. Special Operations Command (USSOCOM) and administered by the U.S. Army Simulation, Training and Instrumentation Command (STRICOM).

The project conducted an HLA requirements analysis, designed and implemented a Run-Time Infrastructure (RTI) connectivity solution, and designed and executed a SOF federation. The requirements analysis involved a study of the Talon training devices and development of a migration plan to convert these devices from the Distributed Interactive Simulation (DIS) protocol to become HLA interoperable. The results of the analysis recommended the incorporation of an HLA middleware solution into each CMS. The effort also involved upgrading the SOF Federation Object Model (FOM). That FOM had been developed for the AC-130U Battle Management Center simulator and later upgraded to include the MH-47E and MH-60K Combat Mission Simulators. The MC-130E and MC-130H SOMs and the SOF FOM are based on the Real-Time Platform Reference FOM (RPR-FOM) version 0.5.

As part of the overall site migration plan, the other DIS devices at the 19th SOS, including the radios, ModSAF, Stealth, and other support devices were interfaced via the STRICOM Gateway to permit interoperability with the HLA Talon devices. As HLA versions of the DIS products and support devices at the 160th SOAR (A) become available, the DIS versions can be replaced by their HLA versions.

The HLA migration of the MC-130E and MC-130H Talon devices was completely successful, with the simulators passing HLA Compliance Testing on July 11, 1999. An Acceptance Test Procedure (ATP) for the Talon devices was performed to verify the HLA implementation. The ATP showed satisfactory performance of the Talon devices, with no perceivable deficiencies. The Special Operations Forces have achieved a significant milestone with the HLA Certification of their Talon devices.

1. INTRODUCTION

The High Level Architecture was established by the Department of Defense (DoD) as an infrastructure that will support simulation reuse and interoperability within the Modeling and Simulation community. The intent of the HLA is to reduce the cost and time associated with constructing individual simulators and groups of interacting simulators, as well as to provide an infrastructure that will permit the development of highly complex simulations.

The Special Operations Forces realized the need and benefit of HLA, and have taken a leadership role in pursuing the use of this new technology for distributed simulation. Their support of the HLA has led to HLA Certification of the MC-130E and MC-130H Combat Talon simulators at the 19th Special Operations Squadron (SOS) at Hurlburt Field, FL. The migration of the Talon devices from DIS to HLA has continued to pave the way for the SOF to fully realize the potential of distributed simulation for training and mission rehearsal.

1.1 Purpose

The purpose of this final report is to document the ADST II effort in performing the HLA migration effort of the MC-130E and MC-130H Talon devices. The report documents the analysis of migrating the legacy devices, the development and integration efforts associated with the HLA implementation, and lessons learned during the migration effort.

1.2 Contract Overview

The Statement of Work (SOW) defined the overarching objectives of this effort as implementing a SOF Federation using the Federation Development and Execution Process (FEDEP). The SOW defined the following objectives of the delivery order:

- Conduct a requirements analysis to determine the best HLA migration path for the MC-130E and MC-130H Talon devices.
- Develop and execute a detailed implementation plan for MC-130E and MC-130H Talon devices for HLA compliance and testing in accordance with the HLA Rules, Interface Specification, and the Object Model Template (OMT).
- Identify the functional requirements for the development of Simulation Object Models (SOM) for the MC-130E and MC-130H simulators.
- Design an HLA Federation in accordance with the HLA FEDEP that meets the needs of the SOF community and is based upon a SOF mission scenario approved by the SOF community.
- Develop and/or revise the necessary SOMs and Federation Object Model (FOM) in accordance with the HLA OMT that are required to execute an exercise at Hurlburt Field with long haul networking to the Ft. Campbell facility.

- Implement the RTI connectivity solution based upon the results of the requirements analysis effort and in accordance with the HLA Rules and Interface Specification.
- Coordinate with other HLA delivery orders to prove the value of HLA tools, infrastructure and process in an integrated development activity on ADST II.
- Investigate and provide recommendations for the replacement of the Plan View Display (PVD), the Stealth, and the Data Logger currently at the Hurlburt training site.
- Complete site integration and HLA compliance testing using the MC-130E and MC-130H baseline software at Hurlburt.
- Interface the Hurlburt Field supporting systems (ModSAF, and the ASTi radios) to the HLA network.
- Design an HLA Federation that will include long haul integration of the federates at Hurlburt (MC-130H, MC-130E, and MC-130U, and the federates at Ft. Campbell (MH-47E and the MH-60K).

The SOW specified that the following assumptions were to be used in planning this effort:

- Subject Matter Experts (SMEs) will be active participants to expedite identification of the functional requirements, evaluation of designs and review of all HLA products.
- All efforts associated with developmental activities will be conducted at the ADST II base facility in Orlando, Florida and at the 19th SOS facility in Hurlburt Field.
- All efforts associated with HLA developmental activities for the MC-130E/H MRD and WST will have been tested and completed prior to integration with Ft. Campbell for HLA long haul networking.
- The Government will assist in the allocation of time and priority in scheduling simulator time, aircrews, and any development efforts required to support the efforts of this SOW.
- An Integrated Product Team (IPT) approach will be used during this effort. The IPT will consist of contractor, military, academia, and government participants and will continuously leverage from the efforts on other delivery orders.
- All software products and commercial off-the-shelf (COTS) hardware will be Year 2000 compliant.

The SOW also specified that the following Government Furnished Property/Equipment would be provided on an as required basis at the ADST II Orlando facility by the Government to support the execution of this program:

- ADST II developmental systems from all previous/on-going delivery order efforts.
- ADST II HLA Testbed resources.

2. APPLICABLE DOCUMENTS

2.1 Government Documents

The following Government documents formed part of the SOW.

- HLA Rules Version 1.3 (DMSO)
- HLA Interface Specification Version 1.3 (DMSO)
- HLA Object Model Template Version 1.3 (DMSO)
- HLA Federation Development and Execution Process (FEDEP) Model Version 1.1 (DMSO)

2.2 Non-Government Documents

The following documents were generated under this Delivery Order. They are available from the ADST II Document Repository.

- Version Description Document, ADST-II-CDRL-MC130-2000049
- Talon Simulator Emulator Version Description Document, ADST-II-CDRL-MC130-2000050
- Network User's Manual, ADST-II-MISC-MC130-2000051
- Talon HLA Cold Start Procedures, ADST-II-MISC-MC130-2000052
- MC130E Simulation Object Model, ADST-II-MISC-MC130-2000053
- MC130H Simulation Object Model, ADST-II-MISC-MC130-2000054
- Special Operations Forces (SOF) Federation Object Model, ADST-II-MISC-MC130-2000055
- HLA Training Presentation, ADST-II-MISC-MC130-2000056
- HLA-IU Engineering Unit Test Procedures, ADST-II-MISC-MC130-2000057
- HLA Interface Acceptance Test Procedures, ADST-II-MISC-MC130-2000060

3. Technical Overview

To achieve an HLA solution for a simulator, there are two core tasks that must be completed. One of these core tasks is the development of a Run-Time Infrastructure connectivity solution for the simulator. This allows the simulator to communicate with the RTI and thus, with other members of the federation.

The other core task is the development of HLA object models, namely the SOM and FOM. The SOM, required as part of the HLA Rules, describes the intrinsic capabilities that the simulator can offer to the HLA federation. The FOM, also required as part of the HLA Rules, provides the information model contract for all members of the federation.

3.1 RTI Connectivity Analysis

The IPT performed an analysis of the different approaches that may be used to transition each CMS from DIS to HLA. These approaches included a DIS-HLA gateway approach, a middleware approach, and a native approach.

3.1.1 Gateway Approach

The gateway approach involves adding a device between the DIS and HLA networks to convert or translate DIS PDUs into corresponding HLA service calls and data. The gateway device also performs the reverse operation and converts HLA data into corresponding PDUs. This method permits unmodified DIS devices to interact with HLA devices. While the gateway approach can be very cost effective, it does have limitations. These include the addition of latency, limited scalability, and that the gateway does not take full advantage of the features of HLA (such as Data Distribution Management or Ownership Management).

3.1.2 Middleware Approach

The middleware approach involves the integration of a commercially available software product into the simulation device. The middleware handles all interfaces to the RTI and network, and typically comes with an application program interface (API) that the software developer uses in linking the simulator application to the middleware solution. The middleware solution offers low latency, scalability, and potentially full HLA capability. In addition, the middleware product can ease the integration of new releases of RTIs and FOMs, particularly if the middleware product maintains a consistent API between versions. As HLA continues to mature, this can be an important consideration.

3.1.3 Native Approach

The native HLA solution involves the implementation of a direct RTI interface with the simulator device. While this approach offers the least latency and can take full advantage of HLA, it often involves extensive rework of the application, development of new software, and significant integration and test time. This approach is also the most costly and poses the highest engineering risk.

3.1.4 Results of RTI Connectivity Analysis

Following the analysis of the different approaches to achieving RTI connectivity, the IPT developed a migration strategy for the Talon devices. This strategy included the integration of a middleware product into each Talon device, replacing its DIS Interface Unit (DIU) with an HLA Interface Unit (HLA-IU). The approach was similar to that used successfully for the AC-130Usimulator at Hurlburt Field. The solution was the most cost-effective HLA-based solution, and was also achievable within the time frame of the contract.

The IPT evaluated the available COTS middleware products, and chose MäK Technologies' VR-Link product (version 3.4) as the best solution. One key reason for choosing VR-Link was that the MC-130E and MC-130H DIU utilized a previous version of VR-Link, thus facilitating an easier integration approach. Also, VR-Link 3.4 supported RTI 1.3v6 and RPR-FOM 0.5, the versions being used for the SOF Federation. This choice would facilitate interoperability among all the SOF simulators. An additional feature of VR-Link was that MäK offers a maintenance agreement with the VR-Link product that provides technical support and software upgrades for newer versions of the RTI and FOMs as they are released.

For the other DIS devices that support the Talon devices, the IPT decided to utilize a gateway to translate between DIS and HLA. The other RTI connectivity approaches were considered for those devices, but since they were COTS products, another HLA solution was either presently unavailable or would require significant development. As the gateway approach does have some limitations, it must be considered to be an interim solution for HLA interoperability of these products until a native HLA solution is available. As such, the recommended HLA migration strategy includes replacement of the DIS devices (such as simulated radios) with HLA native devices when such devices become available.

The HLA IPT evaluated the available HLA Gateway products and chose the STRICOM Gateway as the best solution. This gateway, developed and maintained by Institute for Simulation and Training (IST), utilized RTI 1.3v6 and RPR-FOM 0.5.

3.2 HLA Object Model Development

The project was required to develop SOMs and a FOM for the Talon devices. The IPT chose the RPR-FOM version 0.5 as the target reference FOM to begin development of these object models. The RPR-FOM is a reference FOM that was developed to meet the needs of the real-time platform (and former DIS) community. The long-term migration plan for the SOF community involves the development of a common Special Operations Forces FOM (SOF FOM). The RPR-FOM had already been selected to be the basis of that FOM, for several reasons. First, it is a well thought out FOM for real-time simulators. Second, it can be easily augmented with additional capabilities as needed. Third, it was the only FOM supported by the STRICOM gateway and VR-Link. Fourth, being DIS legacy simulators, the use of the RPR-FOM was recognized as the quickest approach to achieving an HLA compliant solution for the CMS devices. Fifth, the AC-130U BMC at Hurlburt Field and the MH-47E and MH-60K simulators at Ft. Campbell already used the RPR-FOM 0.5 as the basis for their SOMs and FOM. Interoperability with those devices was a requirement of this project, and to achieve interoperability in HLA, all simulators in the same federation must use the same FOM and identical .fed files.

The RPR-FOM contains a foundation set of interoperable objects, attributes, and interactions in Object Model Template (OMT) compliant format. There is also a guidance document referred to as the Guidance, Rationale, and Interoperability Modalities (GRIM) [1] for the RPR-FOM. The set of object model data provided by the RPR-FOM, along with its associated guidance document, eases the task of developing HLA compliant, interoperable federates for both legacy simulators and new simulators. In particular, the effort involved in developing SOMs and a FOM for a federate/federation is shortened considerably.

3.2.1 SOM Development

The SOM development process used for the Talon devices utilized a combination of the Object Model development guidance, the RPR-FOM, the GRIM, and the MC-130E and MC-130H simulator system documentation. The Object Model development guide provided a step by step process to SOM and FOM development. The process includes determining the publication and subscription capabilities for object and interaction classes, as well as their respective attributes and parameters. Once these capabilities are determined, the Object Model Template can be populated with this information.

The process was simplified by the use of the RPR-FOM, which already defined all object and interaction class names, along with their respective attributes and parameters. In addition, all abstract classes and the class subscription hierarchy are defined by the RPR-FOM, as are all enumerated and complex data types and the object model lexicon. The process simplified down to identifying the elements of the RPR-FOM that were supported by the CMS devices, and mapping the simulators' internal data definitions to the RPR-FOM. This mapping was greatly facilitated by the GRIM and the Talon system documentation.

The GRIM provides a mapping between DIS PDUs and the components of the RPR-FOM. Since the Talon devices were DIS simulators, the data model of the simulators' network interface closely matched that of DIS. Using the Talon system documentation and the GRIM, a mapping was made between the data (variables) within the DIU and the RPR-FOM elements. The Talon system documentation included the PDUs transmitted/sent (or published) by the Talon devices, received and utilized (or subscribed to) by the Talon devices, and the respective PDU fields that were supported by each Talon device. This system documentation was used extensively throughout the SOM development process to determine the SOM publication and subscription capabilities.

Once the mapping was completed, the OMT was populated using the DMSO provided Object Model Development Tool (OMDT). This was accomplished by beginning with the complete RPR-FOM 0.5 loaded into the OMDT, and removing the unsupported object classes, interaction classes, attributes, parameters, and enumerated/complex data types. The publication and subscription capabilities were then assigned to the remaining elements.

3.2.2 FOM Development

The long-term goal of the IPT is to develop a SOF FOM that will extend the RPR-FOM to include SOF unique object model elements not contained within the RPR-FOM. However, during the duration of this program, the gateway and middleware products only supported the use of the RPR-FOM, thus limiting us to use this Reference FOM as the basis for our FOM.

For this effort the SOF FOM that had been developed for the AC-130U simulators needed to be modified to publish IFF object attributes. Under RPR-FOM 0.5, IFF was not implemented. In collaboration with IST and the RPR-FOM Standards Development Group, an IFF object representation was developed. The SOM and FOM were updated to include this change, and the simulator interfaces were developed to support this implementation.

3.3 Host Emulator

3.3.1 Introduction

The Talon Host Emulator was developed by reusing and modifying the Host Emulator that had been developed for the AC-130U. The purpose of the Host Emulator was to provide a means to simplify and speed up testing of the HLA-IU software on the HLA Testbed. The emulator does this by allowing a user to place data into simulated shared memory locations, as if it were a Talon host writing to its own shared memory. The HLA-IU can then access this data in the same way it would access the Talon shared memory.

Additional Host Emulator functionality includes a primitive flight simulation routine. This routine provides constantly updated ownship location/movement information to the appropriate memory locations, in such a manner as to simulate a simple, circular, flight path. The Host Emulator also allows the HLA-IU to place remote entity data into the simulated Talon shared memory, where it can be examined through the user interface.

3.3.2 Software Operation

There are two primary aspects of the Host Emulator. The first is the Graphical User Interface (GUI), which is activated when the user enters the program name "emulator" at the UNIX prompt (and while in the appropriate directory). The second is the Initialization file, which will generally be constructed in the same directory.

The GUI consists of 5 distinct regions: Title Bar, Screen Selection Area, Data Entry Area, Command Button Line, and Message Area. In addition, a copy of the operator's manual is provided as a help text popup area. Each of these areas, and the capabilities of each, are described in the operator's manual.

A capability for reading initial values from a text data file has been included. To utilize this capability it is first necessary to create the text file, consisting of a list of variable names (as seen on the data entry screen) with an associated value. More details are available in the online operator's manual.

3.3.3 Software Overview

3.3.3.1 Graphical User Interface

The user interface was written in an X-Windows style, using Motif & X-Toolkit meta-calls. All of the Xwindows setup is performed in the "main" and "add_???" routines. The series of "CB_????" (CallBack) routines are used to process any mouse or keypad inputs to the GUI area.

In order to rapidly switch between the Data Entry Areas (when one is selected in the Screen Selection Area), it was found that the easiest method was to pre-create each screen at startup.

Then we would just use the XtManageChild & XtUnmanageChild calls to make only one screen active at a time.

3.3.3.2 Emulator to HLA-IU Interface

In the actual Talon devices, shared memory is used to pass data between the Talon Host computer and the HLA Interface Unit. For HLA, we retained the same shared memory data structure that was used for the DIS Interface Unit. The Host Emulator uses the UNIX interprocess shared memory functions "shmget" and "shmat". In order for this to work, both the Host Emulator and the HLA-IU must be running on the same computer. The first process running allocates a block of shared memory, and writes the address of this block to a text file, so that the second process can attach to it.

To avoid overflowing the UNIX memory with unreleased blocks, a destructor routine removes the shared memory block (using "shmctl") and the "scraddr.txt" file, whenever one of the processes exits. Because the memory is not actually released until all attached processes become unattached, it is necessary to be sure both processes are down, before restarting either. Otherwise, restarting the down process will cause a new section of UNIX shared memory to be allocated, when it doesn't find the "scraddr.txt" file.

3.3.3.3 Ownship Movement

The ownship movement algorithm is implemented by taking a circle and dividing it by the number of desired iterations, and determining the position of a point on that circle for each iteration. The only non-intuitive aspect of this functionality is that the heading of the vehicle (OWNYAW) appears to be based as if the vehicle was traveling backwards, in reverse [2π - (angle from circle origin to vehicle)].

3.3.3.4 Initialization Data File

The initialization data file is implemented by reading in a text file, and interpreting and writing the data items to the shared memory. The "process_data_to_mem" function is written in a generic manner so that it can be used from several areas.

3.4 STRICOM Gateway

The STRICOM Gateway version 3.2 was utilized to allow the ModSAF 5.0 and other DIS native devices to interoperate with the HLA Talon devices. The Gateway proved to be very robust. The STRICOM Gateway 3.2 was installed in each Talon SGI Origin along with the HLAIU. The 3.2 version was customized to handle IFF for the Talon devices and Light States the Ft. Campbell CMS devices. All SOF sites have the same STRICOM Gateway version.

The STRICOM Gateway version 3.2 was modified by the Institute of Simulation and Training (IST) software developers to adapt the following two requirements:

- The Talon baseline published IFF data under DIS version 6 PDU's, in addition to all other data in DIS version 4 PDU's. Since the DIS network contained two different versions of DIS PDU's, the Gateway needed to convert IFF data published in the HLA network, and create a PDU with a version 6 in the PDU header, thus outputting two types of DIS version PDU's and maintaining the same functionality.

- Two of the HLA federates that participate in the SOF federation, the MH-47E and MH-60K simulators, contained a new set of attributes under the MilitaryAirLandPlatform class that did not exist in the appearance field of the DIS standard. The HLA integration team extended their SOM's to include the following attributes:
    AntiCoLandingLightsOn
    NavigationLightsOn
    NavLightFlashingOn
    NavLightBrightOn
    AntiCollisionLightsOn
    AntiColLightIntensitylUpperLightOn
    AntiColLowerLightOn
    FormationLightsOn
    FormationLightIntensity
    FormationLightNVGOn
    SpotLightsOn
  As a result, the FOM was extended to include these new attributes. To maintain the same version of the STRICOM Gateway among all SOF networks, the Gateway was modified to include these attributes. Note that since these data attributes are not part of the DIS standard, no DIS applications will know what these appearance fields mean. The functionality is in place for any DIS application to process these attributes in the future.

STRICOM Gateway is ideally installed on a workstation with two ethernet cards, with one card designated for DIS traffic and the other designated as the HLA network card. With RTI 1.03 the workstation had to be configured such that the HLA ethernet card was the system's primary ethernet port, while the DIS card is designated as the secondary ethernet port. Failure to do so would prevent BEST EFFORT traffic from being received by the Gateway. With RTI 1.3, which used for the Talon devices, the HLA network card can now be selected in the RTI.rid file. This eliminates the requirement of making the HLA ethernet card the system's primary ethernet port.

Some configuration issues must also be set before the gateway can be used. These options are set in the Sim.cfg file, as explained in the Gateway User Guide. These options must be set to specify site, host, DIS exercise ID, and UDP ports. The DIS IP address is set here, and the network interface card used for the DIS network is also specified here.

Aside from the standard HLA network configuration, i.e., setup of host tables and RTI .rid file, it is important to note the requirements of the Gateway regarding the .fed file. While the RTI uses the .fed file, it is also utilized by the Gateway for initialization functions. Because of this, all classes defined within the provided Gateway .fed file (Giant.fed) must be included within the federation .fed file, regardless of whether or not these classes are part of the FOM.

Thus, the federation .fed file was created by utilizing the Gateway .fed file as a baseline and adding additional classes required by VR-Link.

The gateway was typically run using the autostart command, which automatically invokes a sequence of RTI interface commands for creating the federation, joining the federation, publication declarations, and subscription declarations. Using the autostart command, the gateway subscribes to all classes of the RPR-FOM (as well as the additional classes defined by the Gateway). While the declaration management services are being invoked, the autostart does not effectively utilize declaration management for interest management filtering. An alternative is to write a script and utilize the available gateway commands for declaring the publication capabilities and subscription interests of the gateway federate.

3.5 ASTI Radio

The ASTi Digital Audio Communication System (DACS) resides on the DIS network and communicates voice information through radio PDUs. The DACS system is a COTS product and a native HLA solution was not available. The radios were left as DIS devices, so the functionality and operation of the radios remain the same. The 160th can consider upgrading to HLA radios when such devices become commercially available.

4. SYSTEM DESCRIPTION

4.1 HLA Interface Unit Conceptual Overview

The HLA Interface Unit is a software application that processes the SOF FOM data. The HLA-IU resides on a Silicon Graphics Inc. Origin Indigo 2 computer separate from the Host, and communicates with the Host through shared memory, consisting of a SCRAMNET connection.

Functionally, the HLA-IU provides two services for the Host, publication and subscription of FOM data. The HLA-IU subscribes to SOF FOM data and places it into shared memory for the Host to read and process. The HLA-IU then reads data from the Host and publishes it as SOF FOM data to the network. All SOF FOM data subscription and publication functions supported are in accordance with RTI1.3v6 HLA specifications.

Custom software makes bidirectional data transfers between the local shared memory and the RTI thru VR-Link, which is a COTS HLA API. VR-Link packages outgoing data, dead-reckons entity data, provides smoothing and filtering, and makes most of the RTI service calls to the local RTI component (LRC).

The following diagram shows the concept of placing a HLA Interface Unit (HLA-IU is depicted as the shaded region) between the Host and the HLA network. In accordance with HLA rules, all network communication occurs between the HLA-IU and the RTI.

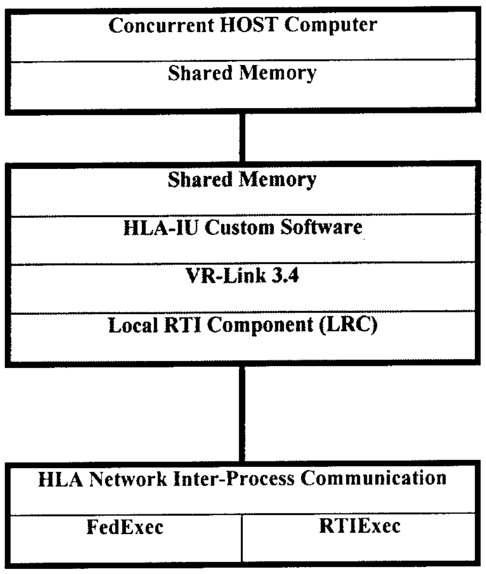

Figure 1. HLA Interface Block Diagram

4.1.1 RTI 1.3v6

The HLA component of the HLA-IU is the Run-Time Infrastructure (RTI) version 1.3 release 6. The RTI serves as the data distribution mechanism in HLA. It provides communication between the HLA-IU and the Local RTI Components (LRC's) of other applications or federates.

The RTI was downloaded from the Defense Modeling and Simulation Office (DMSO) at http://hla.dmso.mil/. This software package includes installation software, release notes, and several source-code examples illustrating much of the API functionality. This software must be installed on and configured for every federate participating in a federation.

The RTI software was installed in the directory path */usr/people/hlaiu/rti1.3v6.*

4.1.2 VR-Link 3.4

The VR-Link Networking Toolkit version 3.4 from MäK Technologies is an object-oriented library of C++ functions and definitions that are used to communicate between the HLA-IU custom software and the RTI. The VR-Link 3.4 software was installed in the directory path */usr/people/hlaiu/vrlink3.4.*

The following are VR-Link's features to speed the creation and maintenance of HLA applications:

4.1.2.1 Exercise Connection Class

The DtExerciseConn class is the exercise connection and serves as the application's interface to the RTI. It is the object through which a VR-Link application connects to the network. VR-Link will create an instance of this class, called *exConn.* Through this object VR-Link can send and receive simulation data. This object provides an efficient way to direct incoming data to other parts of the HLA-IU. Its member functions allow an application to send interactions to the exercise, read input from the network, generate event IDs, and register call back functions.

4.1.2.2 Object Management Classes

To better manage state information, VR-Link informs other exercise participants about states of its locally generated objects, such as entities and emitter systems. In turn, VR-Link receives, processes, and manages state information about remote objects. Object Management classes maintain state information of local and remote objects, and handle the sending and receiving of state updates. These classes provide an entity list to keep track of participants in the federation, and facilitate dead reckoning, trajectory smoothing and filtering.

Entity state and network entity data is maintained by the following VR-Link C++ container classes:

DtEntityPublisher, DtEmitterSystemPublisher, DtEmitterBeamPublisher

> These publisher classes send state update messages of its locally simulated objects to other exercise participants

DtEntityStateRepository, DtEmitterSystemRepository, DtEmitterBeamRepository

> These storage classes store the locally or remotely simulated object's state information. They contain functions for inspecting and changing the components of an object's state.

MyEntityStateRep

> The MyEntityStateRep class is used to bypass the DtEntityStateRepository class when storing a locally simulated entity's state data. This is done when adding or extending the FOM attributes.

DtReflectedEntityList, DtReflectedEmitterSystemList, DtReflectedEmitterBeamList

> VR-Link maintains a list of remote objects in linked lists. Each object known to the list has a DtReflectedEntity, DtReflectedEmitterSystem, or DtReflectedEmitterBeam object to represent it.

Reference the VR-Link User's Guide for a detailed description of each container class.

4.1.2.3 Interaction Classes

Interactions are one-time events such as weapon fire or munition detonations. VR-Link manages interactions through classes derived from DtInteraction, such as DtFireInteraction and DtMunitionDetonationInteraction.

When an interaction occurs on the network, VR-Link receives notification through the exercise connection. The function DtExerciseConn::drainInput() reads and processes the input. The HLA-IU custom software reacts to the interaction through the use of callback functions.

Interactions created by the Host are passed to the HLA-IU custom software, which packages it using VR-Link container classes, which in turn sends the interaction through the exercise connection.

4.1.3 HLA-IU Custom Software

The HLA-IU custom software is an adaptation of the DIS Interface Unit (DIU) already in place from a previous delivery order. The goal of our development plan was to leave the shared memory data structure intact and convert existing DIS class functions to HLA class function calls. In addition, the developers reorganized the source code for improved reading and use of the C++ programming language.

The HLA-IU custom software was installed in the directory path */usr/people/hlaiu/*.

4.1.3.1 The HLA-IU Software Process

The following flowchart is a high level diagram of the lifecycle of the HLA-IU custom software. Following the flowchart is a brief description of each step in the process.

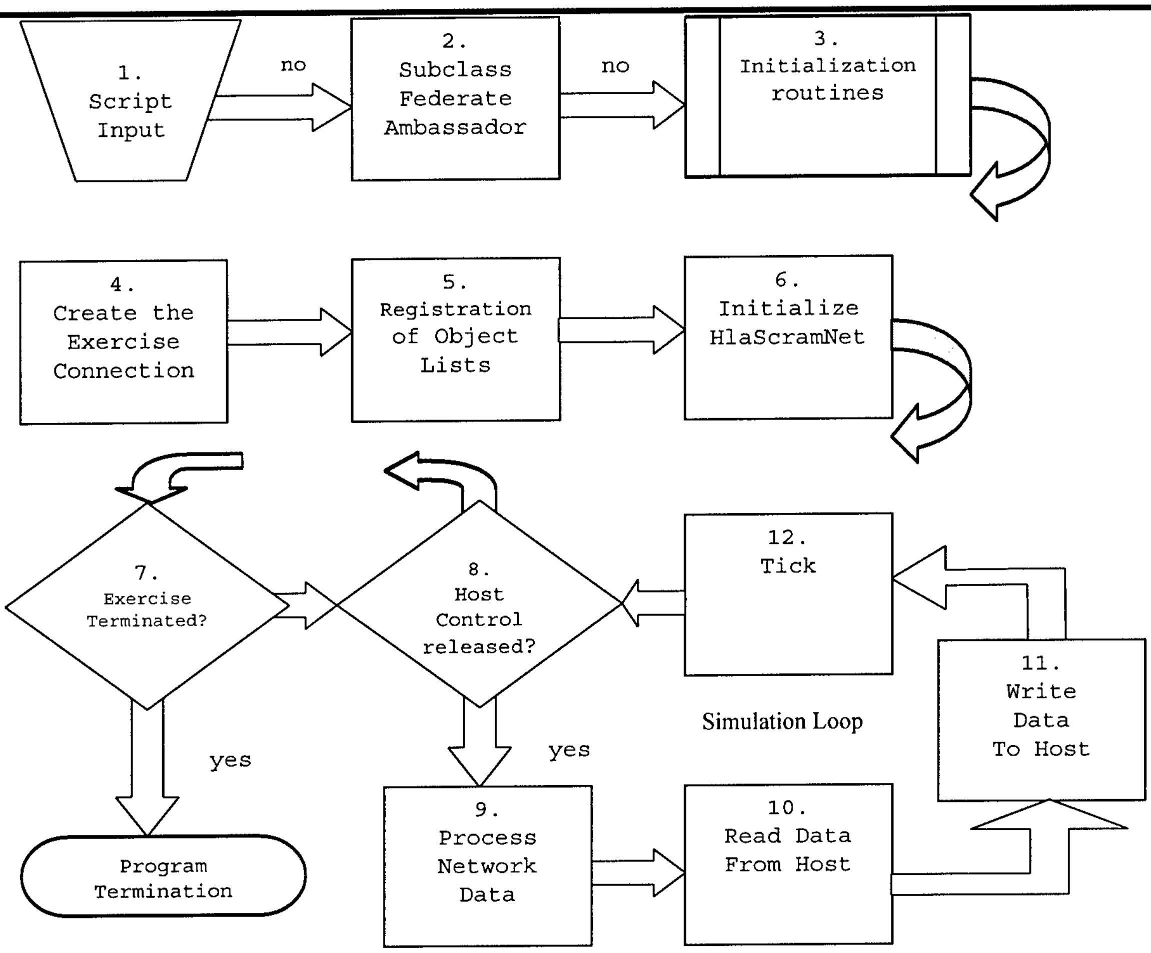
1. Script Input
no
2. Subclass Federate Ambassador
no
3. Initialization routines
4. Create the Exercise Connection
5. Registration of Object Lists
6. Initialize HlaScramNet
7. Exercise Terminated?
8. Host Control released?
12. Tick
11. Write Data To Host
Simulation Loop
yes
yes
Program Termination
9. Process Network Data
10. Read Data From Host

Figure 2. HLA-IU Software Flowchart

1. Script Input

The HLA-IU custom software is launched from the Toolchest menu. The Toolchest menu button on the SGI O2 computer will remotely launch the HLA-IU executable on the SGI Origin 2000 computer. The executable name for the application is *hla*, and is located in the /usr/people/hlaiu/src directory.

2. Subclass the Federate Ambassador

A requirement of the Simulation Object Model (SOM) for the Talon devices was the ability to publish IFF object attributes. The HLA integration team had two options to extend the FOM and SOM to include this object class. First, we could bypass the VR-Link API by obtaining a federate ambassador pointer to the RTI. In this manner we can make all the necessary service calls to the RTI. The second option was to use VR-Link's FOM extension capability. The HLA integration team chose the first option, to bypass VR-Link. This gave the integration team an opportunity to experience a native implementation of an HLA object class. The second option was used on the Ft. Campbell training devices during the implementation of light state attributes. Both methods of FOM/SOM class extensions were successful, and required the same level of effort.

To have access to the RTI services, the class MyFedAmb was derived from DtVrlFederateAmbassador, and new definitions for virtual functions were implemented to register, publish, unpublish, and delete IFF object data.

3. Initialization Routines

Several initialization routines are called during the startup process. DtTimeInit() initializes the simulation time so that all entities are dead-reckoned based on the same value of current time. InitDebug() initializes a command line input parser menu, which allows the user to view specific network data. InitMtl() reads and processes the runtime data defined in the *mtl* file.

4. Create the Exercise Connection

DtExerciseConn is the object through with a VR-Link application connects to the network. Data is sent and received through the exercise connection. Three arguments are passed to the DtExerciseConn constructor. *ExecName* is defined in the *mtl* file, and refers to the federation name (sofhla). *FedName* is also defined in the *mtl* file, and refers to the federate name (MC130E or MC130H). The third parameter specifies the version of the RPR-FOM it plans to use(0.5).

5. Registration of Object Lists

The class MyEntityStateRep registers objects it will publish. The object instance *emt_sys_list* of class DtReflectedEmitterSystemList is registered for subscription of Emitter Systems. The object instance *remotes_list* of class type DtReflectedEntityList is registered for subscription of the following object classes:

BaseEntity.PhysicalEntity.MilitaryEntity.MilitaryPlatformEntity.MilitaryAirLandPlatform

BaseEntity.PhysicalEntity.MilitaryEntity.MilitaryPlatformEntity.MilitaryLandPlatform

BaseEntity.PhysicalEntity.MilitaryEntity.MilitaryPlatformEntity.MilitarySeaSurfacePlatform

BaseEntity.PhysicalEntity.MilitaryEntity.MilitaryPlatformEntity.MilitaryMultiDomainPlatform

BaseEntity.PhysicalEntity.MilitaryEntity.MunitionEntity

BaseEntity.PhysicalEntity.MilitaryEntity.Soldier

6. Initialize HlaScramNet Object

The custom defined C++ class HlaScramNet maintains the process of passing the information between VR-Link and the Host.

The instance *mc130* of class HlaScramNet represents all the properties of reading and writing to shared memory. Private and public member functions process the data through the simulation loop. Several important steps are done during initialization of the *mc130* object, including:

- Pointers to shared memory are allocated (init_scramnet())

- Shared memory is zeroed out (init_arrays())
- Databases are read and stored for processing
- Range filtering is determined
- Determine which device is represented (readOwnship())
- Site/Host/ID is read and assigned
- DIS enumerations for locally generated entities are assigned
- RTI ambassador pointer is allocated
- Entity Publisher is initialized
- Interaction callbacks are registered

7. Exercise State Check

This step in the application is the start of the simulation loop. All data processing occurs after this point. At the start of each frame of the simulation loop, the exercise is polled for termination. If termination has not been cued (by user input of "q" on the parser), processing of data is done for one frame of simulation.

8. Host Control Processing

During each simulation frame, the host must relinquish control to the HLA-IU for processing. If the host does not relinquish control, a counter is incremented indicating the HLA-IU is alive, and a tick is sent to the RTI through the function drainInput(). If the host relinquishes control, data is processed.

9. Process Network Data

The function drainInput() is the VR-Link function that reads and processes all network data. The simulation time spent inside this function is the time spent by the VR-Link API. The HLA-IU custom software processing spends the remaining simulation time.

10. Read Data From Host

Data for various object classes are read from shared memory, and are packaged by the following member functions:

- export_local_entities() – This function is used to publish any local host model data besides the host ownship. This member function is defined in the file hla_rd_export_base_ent.cc.
- readBaseEntAttrFromHost – BaseEntity data is read from shared memory and placed into an instance of the DtEntityPublisher class for the ownship. This member function is defined in the file hla_rd_base_ent.cc.

- readCollParamFromHost – Collision data is read from shared memory and placed into an instance of the DtCollisionInteraction class. This member function is defined in the file hla_rd_collision.cc.
- readEmissionsFromHost – Emitter System data is read from shared memory and placed into an instance of the DtEmitterSystemPublisher or DtEmitterBeamPublisher class. This member function is defined in the file hla_rd_emis.cc.
- readIFFfromHost – IFF system data is read from shared memory and sent to the network as updates. This member function is defined in the files hla_create_iff.cc and hla_rd_iff.cc.

11. Write Data To Host

Data is collected from the network during drainInput() and stored in various VR-Link container classes. The data is then retrieved by the following HlaScramNet member functions and sent to the host:

- writeBaseEntAttrToHost – this function is the most time intensive function in the custom software. All network entities are sorted by slant range to determine the closest 40 entities to the host. The sorted entities are then processed so that they are sent to the host shared memory in the same memory slot as for the previous pass. The algorithm is difficult to follow. A simpler implementation using a linked list rather than a bubble sort of an array proved to be more time consuming and was dropped. This member function is defined in the file hla_wr_base_ent.cc.
- writeBaseMunAttrToHost – If fire interactions occur during a simulation frame, this function will package the munition data and send it to the host. This member function is defined in the file hla_wr_base_munition.cc.
- writeMunEmissionsToHost – If any guided munitions contain emitter data during a simulation frame, this function will package the emitter data and send it to the host. This member function is defined in the file hla_wr_mun_emis.cc.
- writeCollisionParamToHost – Collision data collected through an interaction callback is sent to the host shared memory. This member function is defined in the file hla_wr_collision.cc.
- writeWeapFireParamToHost – WeaponFire data collected through an interaction callback is sent to the host shared memory. This member function is defined in the file hla_wr_fire.cc.
- writeMunDetParamToHost – MunitionDetonation class data collected through an interaction callback is sent to the host shared memory. This member function is defined in the file hla_wr_deton.cc.
- writeEmissionsToHost – Emitter System and Beam class data that maps to an existing entity on the sorted list structure (which is built in writeBaseEntAttribToHost()) is

packaged, sorted, and the closest three are sent to the host shared memory. This member function is defined in the file hla_wr_emis.cc.

12. Tick

The member function simTick() is where all object state repositories will send the data through VR-Link to the network.

4.1.3.2 HLA-IU Run-Time Configuration Data

The HLA-IU reads and processes configuration files at runtime. These files can be modified without having to recompile the HLA-IU source code.

4.1.3.2.1 MTL files

An *mtl* file is a MaK Technologies Lisp configuration file. This type of file contains default parameters for the HLA-IU. Two *mtl* files are used, depending on the training device. The MC-130E uses the file hla.mtl.wst, and the MC-130H uses the file hla.mtl.mrd. These files must be copied into the runtime file hla.mtl when the user makes any changes.

The following are the selections that are suitable for changing, thus taking advantage of dynamic exercise planning:

ExerciseId	This option changes the exercise id. The default is 19.
own_marking	This option changes the markings of the training device
AreaOfInterest	This option will take as an integer the slant range in meters that network entities must be from the ownship in order to be processed. Network entities exceeding this slant range will be excluded from processing. The default is 160000 meters, roughly about 100 miles.

The following can be edited, but the HLA integration team discourages this unless the network operator is aware of the implications of making these changes:

execName	This option changes the name of the federation. It takes a string of less than 28 characters.
fedName	This option changes the name of the SOACMS device as listed on the fedEx window. It takes a string of less than 28 characters
SiteId	This option changes the site id of the training device
hmrd_applNum_OWN	This option changes the host id of the MC-130H training device
ewst_applNum_OWN	This option changes the host id of the MC-130E training device
EntityId	This option changes the entity id of the training device
iff_mode_1	This option changes the mode 1 parameter for IFF on the MC-130H only
iff_mode_2	This option changes the mode 2 parameter for IFF on the MC-130H only
iff_mode_3a	This option changes the mode 3a parameter for IFF on the MC-130H

	only
iff_mode_4	This option changes the mode 4 parameter for IFF on the MC-130H only
TimeOut	This option will turn on the timing out mechanisms for objects that are orphaned (i.e. ghost entities) in the HLA network.
TimeOutInterval	This option takes as an integer the time in seconds that should pass before timing out an orphan object on the HLA network.

4.1.3.2.2 Model Mapping Files

The HLA-IU reads three database files that are used to map network model data to host model data.

4.1.3.2.2.1 Model_List

The file Model_List maps the DIS enumeration of a network entity to a model number and emitter model number for the host.

4.1.3.2.2.2 Ece_mapping.txt

The file ece_mapping.txt contains the emitter names for emitter systems. It maps the field value taken from Section 8 of the EBV-DOC referenced in DIS IEEE Std 1278.1-1995 to a Signal Index to pass to the host.

4.1.3.2.2.3 Hlafilter.txt

The file hlafilter.txt is available to the user to insert DIS enumerations for host filtering. Under configuration management, the file is empty.

4.2 Network Overview

Figure 3 depicts the simulator network for the Talon devices at Hurlburt Field. There are two primary Local Area Networks (LAN). The HLA LAN has network connections to the MC-130E HLA-UI, the MC-130H HLA-IU, and the MäK Data Logger. These systems can be used together for HLA simulations that run within the Talon facilities.

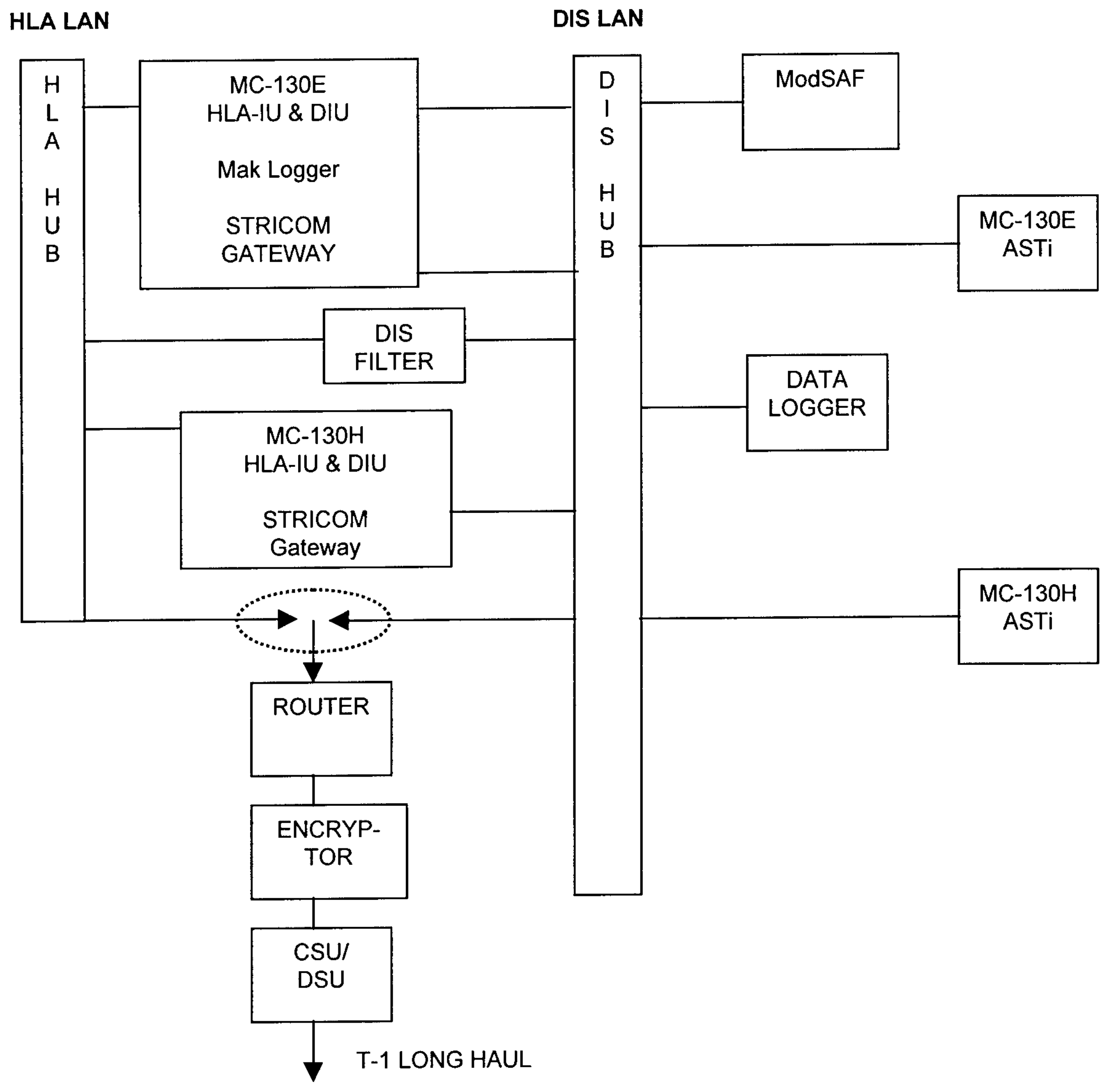

Figure 3. Hurlburt Field Talon Simulator Network

The other primary LAN is the DIS LAN. The DIS LAN has connections to the ModSAF 5.0, the ASTi radios, and the DIS Data Logger. The MC-130E and the MC-130H can connect to the DIS LAN through their respective DIUs. These systems can operate together in DIS for simulations within the Hurlburt facility.

The STRICOM Gateway is connected to both LANs. It is used to translate HLA-to-DIS and DIS-to-HLA for simulations that include both HLA and DIS devices. The STRICOM Gateway is set up to receive 24 of the 27 DIS 2.0.4 Protocol Data Units (PDU) from the DIS LAN, map the data to the SOF FOM, and publish it on the HLA LAN. Likewise it is set up to receive SOF FOM data from the HLA LAN, map it to 24 of the 27 DIS 2.0.4 PDUs, and send it out on the DIS LAN. The three PDUs that it is set to reject are Signal, Transmitter, and Receiver. In other words, the Gateway is set up to process everything except radio related data. Because the Gateway is installed on the same system as the HLAIU both federates use the same network port. This is a unique capability of HLA, which allows multiple Federates running on the same system to share a common Local RTI Component (LRC).

Radio data is processed by the DIS Filter, sometimes also referred to as the XCAIU. The DIS Filter is only needed during HLA long haul exercises with DIS radios. It allows radio data to remain in DIS, be mixed with HLA entity and interaction data, and have both kinds of protocols go out simultaneously over the T-1 line. By letting radio data bypass the Gateway, the Gateway is able to handle more total entities and interactions without degrading its performance.

For HLA long haul exercises, the router's Ethernet cable is connected to the HLA Hub. For DIS long haul exercises, it is connected to the DIS Hub. The router output goes to the Encryptor, which is connected to the CSU/DSU that serves as the high-speed modem for the T-1 line. Locally generated data is thereby sent to a remote location. Likewise, data from a remote location enters the network from the T-1 line, passes through the CSU/DSU, is decrypted, goes to the router, and is output onto one of the LANs.

The STOW-A exercise that was conducted on October 1999 is an example of an HLA long haul exercise with DIS radios. For STOW-A the MC-130E simulator was running in HLA using the HLA-IU. The Gateway performed the bi-directional translations between the HLA LAN and the DIS LAN. The DIS Filter processed the radio traffic, and the router was cabled to the HLA Hub.

5. Technical Discussion and Lessons Learned

5.1 HLA Compliance Testing

The HLA compliance testing process is the means to ensure that DoD simulations are, in fact, HLA-compliant in accordance with DoD policy. The HLA certification process is facilitated through an on-line web-based interface and includes four steps, culminating with the receipt of the HLA compliance certificate. The four steps are as follows: 1) submit the Test Application, 2) submit the completed Conformance Notebook, 3) submit the Test Environment Information, and 4) execute the Interface Test and request theCertification Summary Report.

The MC-130E and MC-130H test applications were submitted, and the conformance notebook data was provided to the DMSO HLA certification agents. This included the SOMs, conformance statements, and scenario data. The conformance statements contain a list of all RTI services supported by the simulators. Since the VR-Link middleware product was used exclusively to interface with the RTI, the RTI services were verified with MäK.

A scenario SOM was generated to be the RepSOM for generating the compliance test sequence. This SOM included the set of object classes, attributes, and interactions necessary to demonstrate the services listed in the conformance statement. The objects and interactions in the scenario SOM were chosen such that they could be easily stimulated during the compliance test.

The test environment information was furnished to the DMSO compliance test agents, and the compliance test agents generated a testable sequence file. The testable sequence file contains a list of services with required object classes, interaction classes, and/or attributes that are required to be implemented as part of the interface test. From the testable sequence file, a procedure was developed to conduct the test. The procedure consisted of a step by step process to invoke the required RTI services and parameters. The required RTI services include two varieties – calls made by the MC-130E and MC-130H applications to the RTI, and RTI callbacks to the MC-130E and MC-130H applications.

The interface tests were conducted at Hurlburt Field on June 23, 1999. All service calls were handled properly, and HLA compliance was achieved for the MC-130E and MC-130H Combat Talon Simulators.

5.2 On-site Integration and Test (Continue)

Development of the HLA-IU was done for the most part on the HLA Testbed at the Operational Support Facility (OSF) using the Host Emulator to debug and test the software as it was being developed. The goal was to perform as much of the testing as possible at the OSF, and to minimize the amount of integration time needed on the Talon devices at Hurlburt Field. The software emulator proved to be an extremely effective way to accomplish this goal. The development used the "build a little, test a little" paradigm. The first major build addressed base entity. The basic HLA-IU framework was built, and the capability for publishing and scribing to base entities was developed. This build was tested on the HLA Testbed using the Host Emulator. It was then taken to Hurlburt and tested on one of the Talon devices. Since both devices are essentially identical from the HLA-IU viewpoint, we could test effectively on either.

The second major build added Emissions and IFF. These were tested on the HLA Testbed and with the Talon devices. We then developed the software to subscribe to Fire and Detonate, and included it in the third major build. Build 3 was tested at the OSF and then at Ft. Campbell. Also during the time frame of Build 2 and 3, we were attempting to establish long haul communications in HLA between the Talon devices at Hurlburt field and the CMS devices at Ft. Campbell.

The on-site software integration effort was greatly assisted by a host software expert. It is highly recommended that the on-site support by a host software expert and/or knowledgeable system operator be part of all future HLA migrations.

The key objectives of system integration were as follows:

- Introduce a separate HLA network by means of a small hub
- Test and debug the network setup
- Install, test and debug each software build

- Conduct and pass HLA Compliance Testing
- Provide a demonstration of the HLA capabilities and training site personnel in how to utilize the new capabilities
- Establish long haul connectivity with Hurlburt, and conduct a joint demonstration in HLA

All of these objectives were successfully accomplished. In addition, the HLA development team, working under the auspices of a different Delivery Order, supported the HLA and long haul aspects of the STOW-A practices, and the STOW-A exercise

5.3 SOF Federation Overview

The current configuration of the SOF Federation is depicted in Figure 4. HLA operates with the following types of components.

- A set of one or more participant applications called federates, participating in a federation
- A run-time infrastructure that is in charge of data distribution, called the RTI
- A process that monitors the status of the federates, called the FEDEX
- A federate that can represent applications that run in the DIS network, called the Gateway.

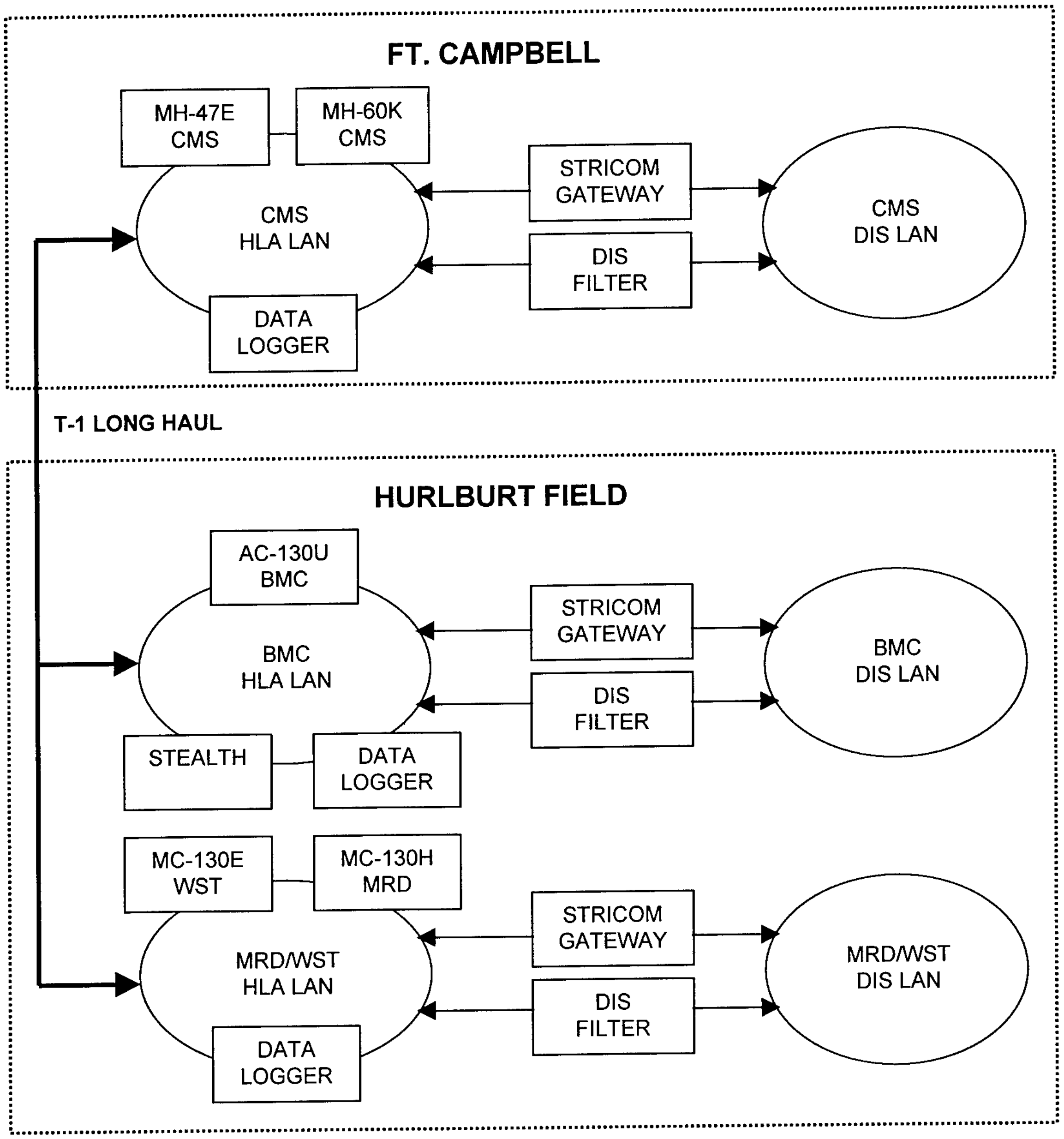

Figure 4. SOF Federation Current Configuration

The RTI version 1.3v6 is the process that coordinates communications between federates. Federates send and receive network data to and from the RTI through a series of service calls, in accordance with the HLA Interface Specification. The RTI must be present on all systems that contain a federate application, and a single RTI process (the RTIexec) must be launched before the start of an exercise. All federates in the federation must have identical RTI versions and identical .fed files.

A federation is simply the network exercise that is running. The federation that utilizes the CMS devices is called the Special Operations Forces Federation. The SOF Federation allows the following HLA devices to participate in an exercise:

- MC-130E WST
- MC-130H MRD
- MRD Gateway
- WST Gateway
- Talon Data Logger
- AC-130U BMC
- BMC Gateway
- BMC Data Logger
- BMC Stealth
- MH-47E CMS
- MH-60K CMS
- Ft. Campbell Gateway
- Ft. Campbell Logger

During a 19th SOS local federation execution, the first nine federates listed above can participate. In the long haul mode of operation, all of the above federates can join the federation and become part of an HLA exercise.

6. Conclusion

The MC-130E and MC-130H Simulators successfully achieved HLA Compliance on June 23, 1999. A demonstration of the simulator interoperating with the MH-47E and MH-60K simulators at Ft. Campbell was used to successfully validate the HLA long haul implementation. The Special Operations Forces have achieved another significant milestone with the HLA migration of the CMS devices. The benefit of this new technology will become even more evident as other SOF devices are migrated to HLA. The HLA will help pave the way for the Special Operations Forces to more fully utilize the potential of distributed simulation technology, including allowing them to more cost effectively develop and utilize more complex training scenarios, and rehearse more complex missions.

7. Points of Contact

SAIC SOFHLA Team

Gilbert Gonzalez	Project Director	407-306-4612
Ivan Carbia	Project/Software Engineer	407-306-4047
Glenn Valentine	Software Engineer	407-306-1083

STRICOM

Tom Smith	Director, STS	407-384-3666
Robert Miller	Project Director	407-384-3685

NAWCTSD

Victor Colon	Project Engineer	407-306-4023

Customer Point of Contact

Maj Mike Vaughn	19th SOS	850-881-2286

STRICOM Gateway

Doug Wood	S/W Lead	407-658-5066

8. Acronym List

ADST II	Advanced Distributed Simulation Technology II
API	Application Program Interface
BMC	Battle Management Center
CDF	Core DIS Facility
CGF	Computer Generated Forces
CM	Configuration Management
CMS	Combat Mission Simulator
COTS	Commercial Off-the-Shelf
DIS	Distributed Interactive Simulation
DIU	DIS Interface Unit
DMSO	Defense Modeling & Simulation Office
DO	Delivery Order
DoD	Department of Defense
FEDEP	Federation Execution and Development Process
FEPW	Federation Execution Planner's Workbook
FOM	Federation Object Model
GUI	Graphical User Interface
HLA	High Level Architecture
HLA-IU	HLA Interface Unit
IPT	Integrated Product Team
IST	Institute for Simulation and Training
LAN	Local Area Network
LRC	Local RTI Component
ModSAF	Modular Semi-Automated Forces
MRD	Mission Rehearsal Device

OMDT	Object Model Development Tool
OMT	Object Model Template
OTB	OneSAF Testbed
OSF	Operational Support Facility
PDU	Protocol Data Unit
PVD	Plan View Display
RPR-FOM	Real-Time Platform Reference FOM
RTI	Runtime Infrastructure
SGI	Silicon Graphics, Inc.
SOF	Special Operations Forces
SME	Subject Matter Expert
SOAR (A)	Special Operations Aviation Regiment (Airborne)
SOM	Simulation Object Model
SOW	Statement of Work
STRICOM	Simulation, Training, and Instrumentation Command
USSOCOM	US Special Operations Command
WST	Weapon System Trainer

APPENDIX A
MC-130E AND MC-130H SOM

Use the following document in CM:

MC130E Simulation Object Model, ADST-II-MISC-MC130-2000053

MC130H Simulation Object Model, ADST-II-MISC-MC130-2000054

Note: This document can be viewed using the Object Model Development Tool 1.3 from DMSO.

APPENDIX B
SOF FOM 0.5

Use the following document in CM:

Special Operations Forces (SOF) Federation Object Model, ADST-II-MISC-MC130-2000055

Note: This document can be viewed using the Object Model Development Tool 1.3 from DMSO.

APPENDIX C
CONFORMANCE STATEMENT

```
createFederationExecution YES
destroyFederationExecution YES
joinFederationExecution YES
resignFederationExecution YES
registerFederationSynchronizationPoint NO
confirmSynchronizationPointRegistration NO
announceSynchronizationPoint NO
synchronizationPointAchieved NO
federationSynchronized NO
requestFederationSave NO
initiateFederateSave NO
federateSaveBegun NO
federateSaveComplete NO
federationSaved NO
requestFederationRestore NO
confirmFederationRestorationRequest NO
federationRestoreBegun NO
initiateFederateRestore NO
federateRestoreComplete NO
federationRestored NO
publishObjectClass YES
unpublishObjectClass NO
publishInteractionClass YES
unpublishInteractionClass NO
subscribeObjectClassAttributes YES
unsubscribeObjectClass NO
subscribeInteractionClass YES
unsubscribeInteractionClass NO
startRegistrationForObjectClass YES
stopRegistrationForObjectClass YES
turnInteractionsOn YES
turnInteractionsOff YES
registerObjectInstance YES
discoverObjectInstance YES
updateAttributeValues YES
reflectAttributeValues YES
sendInteraction YES
receiveInteraction YES
deleteObjectInstance YES
removeObjectInstance YES
localDeleteObjectInstance NO
changeAttributeTransportationType NO
changeInteractionTransportationType NO
attributesInScope NO
attributesOutOfScope NO
requestAttributeValueUpdate YES
provideAttributeValueUpdate YES
turnUpdatesOnForObjectInstance YES
turnUpdatesOffForObjectInstance YES
unconditionalAttributeOwnershipDivestiture NO
negotiatedAttributeOwnershipDivestiture NO
requestAttributeOwnershipAssumption NO
attributeOwnershipDivestitureNotification NO
attributeOwnershipAcquisitionNotification NO
```

```
attributeOwnershipAcquisition NO
attributeOwnershipAcquisitionIfAvailable NO
attributeOwnershipUnavailable NO
requestAttributeOwnershipRelease NO
attributeOwnershipReleaseResponse NO
cancelNegotiatedAttributeOwnershipDivestiture NO
cancelAttributeOwnershipAcquisition NO
confirmAttributeOwnershipAcquisitionCancellation NO
queryAttributeOwnership NO
informAttributeOwnership NO
isAttributeOwnedByFederate NO
enableTimeRegulation NO
timeRegulationEnabled NO
disableTimeRegulation NO
enableTimeConstrained NO
timeConstrainedEnabled NO
disableTimeConstrained NO
timeAdvanceRequest NO
timeAdvanceRequestAvailable NO
nextEventRequest NO
nextEventRequestAvailable NO
flushQueueRequest NO
timeAdvanceGrant NO
enableAsynchronousDelivery NO
disableAsynchronousDelivery NO
queryLBTS NO
queryFederateTime NO
queryMinimumNextEventTime NO
modifyLookahead NO
queryLookahead NO
retract NO
requestRetraction NO
changeAttributeOrderType NO
changeInteractionOrderType NO
createRegion NO
modifyRegion NO
deleteRegion NO
registerObjectInstanceWithRegion NO
associateRegionForUpdates NO
unassociateRegionForUpdates NO
subscribeObjectClassAttributesWithRegion NO
unsubscribeObjectClassWithRegion NO
subscribeInteractionClassWithRegion NO
unsubscribeInteractionClassWithRegion NO
sendInteractionWithRegion NO
requestAttributeValueUpdateWithRegion NO
enableClassRelevanceAdvisorySwitch YES
disableClassRelevanceAdvisorySwitch NO
enableAttributeRelevanceAdvisorySwitch YES
disableAttributeRelevanceAdvisorySwitch NO
enableAttributeScopeAdvisorySwitch NO
disableAttributeScopeAdvisorySwitch NO
enableInteractionRelevanceAdvisorySwitch YES
disableInteractionRelevanceAdvisorySwitch NO
```

APPENDIX D
HLA CERTIFICATES
AND
LETTERS

The certificates and letters for both Talon devices are available in hard-copy only.

APPENDIX E
MC-130E AFTER ACTION REVIEW

Simulation Information

Simulation Name:	MC-130E WST, v. 1.3
Simulation Developer's Name:	Mr. Robert Miller
Simulation Developer's Organization:	U.S. Army STRICOM

Contact Information

POC:	William Garbacz
Email:	William.R.Garbacz@lmco.com
TELEPHONE NO:	(407) 306-2310

1. REGARDING HLA COMPLIANCE CERTIFICATION:

A. What do you consider to be the best part of the process?
The web based submission of Conformance Statement, SOM, scenario data is very efficient.
B. What did you most dislike about the process?
The fact that there are no test tools available to pre-test the federate prior to conducting the actual I/F test with AB Technologies.

C. If you could restructure any part of HLA compliance certification, what changes would you make?
Tools should be provided to allow federate developers to pre-test their federates. This would significantly cut down on the amount of time consumed during the actual I/F test.
D. Was the compliance certification easier or more difficult than you anticipated?
X 1. Easier
☐ 2. More difficult

Please explain:
This particular test was conducted within about 15 minutes, as the test executed as planned without many glitches.

2. PLEASE PROVIDE A BRIEF DESCRIPTION OF YOUR SIMULATION APPLICATION: (Was your goal to develop a Federation or an HLA compliant Federate?)

Our primary goal was to develop an HLA compliant federate. However, this federate was one of many SOF simulators undergoing HLA migration. Thus, each simulator interface (defined by it's SOM) was designed to match the requirements of the federation as a whole (defined by the FOM).

This particular federate is a training simulator.

3. BRIEFLY DESCRIBE YOUR HLA MIGRATION STRATEGY:

☐ A. Developed a native HLA application.

☐ B. Transitioned from ALSP to the HLA. If so, describe the transition path and state the target HLA specification:

☒ C. Transitioned from Distributed Interactive Simulation (DIS) protocol. If so, describe the transition path and state the target HLA specification:
☐ D. Other: (Please describe.)

This effort involved transitioning a training simulator from DIS to HLA, using RTI 1.3v6 and RPR-FOM 0.5. The effort included the integration of other federation components necessary to conduct an effective SOF training exercise, including gateways, stealths, and data loggers.

4. BRIEFLY DESCRIBE THE LEVEL OF EFFORT DEVOTED TO THE TRANSITION:

A. Number of staff months devoted to the transition effort. 24

B. Number of personnel assigned to the project. 6

C. Did the cost of transition involve expenses beyond those specifically required for the transition to the HLA? (Examples may include: rewriting of simulation code, purchase of equipment and or upgrade of infrastructure, augmentation of the existing model capabilities, etc.) If so, please describe and provide an estimated level of effort in staff time:
Much of the effort involved the integration of other federation components such as a gateway and stealth. These other federates required fixes and additional capabilities to meet the needs of our federation. The bulk of this time is rolled up under On-site Integration and Test included in the metrics below.

5. WHICH OF THE FOLLOWING TASKS DID YOU PERFORM AS PART OF YOUR HLA TRANSITION? (Check the appropriate boxes)

☒ A. Defined requirements for the Federation as a whole (% of total effort):	2 %
☒ B. Developed a conceptual model for the Federation as a whole (% of total effort):	1 %
☐ C. Developed and recorded a Simulation Object Model in the Object Model Template format (% of total effort):	0 %
☒ D. Designed and built an RTI interface for your Federate (% of total effort):	54 %
☒ E. Participated in Conformance Testing for the Federate (% of total effort):	2 %
☒ F. Produced a Federation Object Model (% of total effort):	2 %
☐ G. Participated in Federation Object Model development sessions (% of total effort):	0 %
☒ H. Modified an existing Simulation Object Model to reflect the Simulation Object Model requirements for the Federation. (% of total effort):	3 %
☒ I. Planned a federation execution (% of total effort):	2 %
☐ J. Completed the Federation Execution Planners Workbook (% of total effort):	0 %
☒ K. Acquired the equipment and network resources necessary to execute the Federation (% of total effort):	1 %
☒ L. Executed the Federation (% of total effort):	1 %
☒ M. Developed Documentation (% of total effort):	12 %
☐ N. Documented the results of the Federation Execution (% of total effort):	0 %
☒ O. Documented the results and experiences of becoming HLA compliant (% of total effort):	2 %
☒ P. Other: (% of total effort): On-site Integration&Test	18 %

6. HOW WOULD YOU CHARACTERIZE YOUR PROGRAMMING EXPERIENCE AND THAT OF YOUR DEVELOPMENT TEAM WITH REGARD TO SIMULATION DEVELOPMENT?

☒ A. High. Extensive experience.
☐ B. Moderate. Some experience, but simulation development is not our core task.
☐ C. Low. This was our first exposure to simulation development.

7. HOW WOULD YOU CHARACTERIZE YOUR LEVEL OF EXPERIENCE WITH REGARD TO THE HLA?

☒ A. High. We have been involved with the program since it's early development.
☐ B. Moderate. We are well versed on the concept and principals, but have never participated in a development effort.
☐ C. Low. Until we got the task, we really hadn't been involved.

8. DID YOU AND OR YOUR DEVELOPMENT TEAM HAVE ACCESS TO OUTSIDE SUPPORT DURING THE TRANSITION?

☒ A. Specialized HLA training sessions (describe):
Some of the development team members have attended regional and hands-on HLA training.

☐ B. HLA Cadre or Mentor Assistance:

☐ C. Commercial Contractor:

9. WERE THERE ANY UNIQUE OBSTACLES THAT HAD TO BE OVERCOME DURING THE TRANSITION PROCESS, IF SO PLEASE DESCRIBE AND PROVIDE AN ESTIMATED LEVEL OF EFFORT IN STAFF TIME?

This effort involved the development and implementation of new HLA object classes and attributes that were not included in the current version of the RPR-FOM. This added an estimated 1-2 man-months worth of effort to implement this within the SOM/FOM and across all applicable federates.

10. CAN YOU CHARACTERIZE THE TECHNICAL APPROACH USED TO ACQUIRE A RUN TIME INFRASTRUCTURE (RTI) INTERFACE FOR YOUR FEDERATE? (Select those that apply)

☒ A. Purchased a commercial product
☐ B. Acquired a Government off the Shelf (GOTS) product or another freeware product
☐ C. Designed and built a middleware application
☐ D. Extended or modified an existing native interface to the Federate
☐ E. Designed and built a native interface to the Federate
☐ F. Used a gateway or other interface device

11. ARE THERE ADDITIONAL ASPECTS OF THE TRANSITION OF YOUR SIMULATION TO THE HLA THAT WARRANT MENTION? IF SO, PLEASE DESCRIBE AND PROVIDE AN ESTIMATED LEVEL OF EFFORT IN STAFF TIME?

The program plan included the development, test, and integration in a test-bed prior to installing the HLA software on-site. This was done to mimimize travel time and on site integration time, and also to minimize simulator down-time to allow for training.

It is also important to mention that this development effort was done in conjunction with several other simulators also undergoing similar HLA migrations. The development and integration time was likely reduced by 1/3 as we took advantage of the synergies among these programs.

The total number of staff months/personnel, as well as the metrics information provided above reflect the effort involved in converting one federate (MC-130E) from DIS to HLA. In reality, the effort was shared among the development of another nearly identical simulator (MC-130H). Thus, the metrics quoted above could have been reduced by 50% to account for this, but this would have been misleading. Instead, the metrics were provided as if there was only

APPENDIX F
MC-130H AFTER ACTION REVIEW

Simulation Information

Simulation Name:	MC-130H MRD, v. 1.3
Simulation Developer's Name:	Mr. Robert Miller
Simulation Developer's Organization:	U.S. Army STRICOM

Contact Information

POC:	William Garbacz
Email:	William.R.Garbacz@lmco.com
TELEPHONE NO:	(407) 306-2310

12. REGARDING HLA COMPLIANCE CERTIFICATION:

A. What do you consider to be the best part of the process?
The web based submission of Conformance Statement, SOM, scenario data is very efficient.
B. What did you most dislike about the process?
The fact that there are no test tools available to pre-test the federate prior to conducting the actual I/F test with AB Technologies.

C. If you could restructure any part of HLA compliance certification, what changes would you make?
Tools should be provided to allow federate developers to pre-test their federates. This would significantly cut down on the amount of time consumed during the actual I/F test.
D. Was the compliance certification easier or more difficult than you anticipated?
X 1. Easier
☐ 2. More difficult

Please explain:
This particular test was conducted within about 15 minutes, as the test executed as planned without many glitches.

13. PLEASE PROVIDE A BRIEF DESCRIPTION OF YOUR SIMULATION APPLICATION: (Was your goal to develop a Federation or an HLA compliant Federate?)

Our primary goal was to develop an HLA compliant federate. However, this federate was one of many SOF simulators undergoing HLA migration. Thus, each simulator interface (defined by its SOM) was designed to match the requirements of the federation as a whole (defined by the FOM).

This particular federate is a training simulator.

14. BRIEFLY DESCRIBE YOUR HLA MIGRATION STRATEGY:

☐ A. Developed a native HLA application.

☐ B. Transitioned from ALSP to the HLA. If so, describe the transition path and state the target HLA specification:

☒ C. Transitioned from Distributed Interactive Simulation (DIS) protocol. If so, describe the transition path and state the target HLA specification:
☐ D. Other: (Please describe.)

This effort involved transitioning a training simulator from DIS to HLA, using RTI 1.3v6 and RPR-FOM 0.5. The effort included the integration of other federation components necessary to conduct an effective SOF training exercise, including gateways, stealths, and data loggers.

15. BRIEFLY DESCRIBE THE LEVEL OF EFFORT DEVOTED TO THE TRANSITION:

A. Number of staff months devoted to the transition effort. 24

B. Number of personnel assigned to the project. 6

C. Did the cost of transition involve expenses beyond those specifically required for the transition to the HLA? (Examples may include: rewriting of simulation code, purchase of equipment and or upgrade of infrastructure, augmentation of the existing model capabilities, etc.) If so, please describe and provide an estimated level of effort in staff time:
Much of the effort involved the integration of other federation components such as a gateway and stealth. These other federates required fixes and additional capabilities to meet the needs of our federation. The bulk of this time is rolled up under On-site Integration and Test included in the metrics below.

16. WHICH OF THE FOLLOWING TASKS DID YOU PERFORM AS PART OF YOUR HLA TRANSITION? (Check the appropriate boxes)

☒ A. Defined requirements for the Federation as a whole (% of total effort):	2 %
☒ B. Developed a conceptual model for the Federation as a whole (% of total effort):	1 %
☐ C. Developed and recorded a Simulation Object Model in the Object Model Template format (% of total effort):	0 %
☒ D. Designed and built an RTI interface for your Federate (% of total effort):	54 %
☒ E. Participated in Conformance Testing for the Federate (% of total effort):	2 %
☒ F. Produced a Federation Object Model (% of total effort):	2 %
☐ G. Participated in Federation Object Model development sessions (% of total effort):	0 %
☒ H. Modified an existing Simulation Object Model to reflect the Simulation Object Model requirements for the Federation. (% of total effort):	3 %
☒ I. Planned a federation execution (% of total effort):	2 %
☐ J. Completed the Federation Execution Planners Workbook (% of total effort):	0 %
☒ K. Acquired the equipment and network resources necessary to execute the Federation (% of total effort):	1 %
☒ L. Executed the Federation (% of total effort):	1 %
☒ M. Developed Documentation (% of total effort):	12 %
☐ N. Documented the results of the Federation Execution (% of total effort):	0 %
☒ O. Documented the results and experiences of becoming HLA compliant (% of total effort):	2 %
☒ P. Other: (% of total effort): On-site Integration&Test	18 %

17. HOW WOULD YOU CHARACTERIZE YOUR PROGRAMMING EXPERIENCE AND THAT OF YOUR DEVELOPMENT TEAM WITH REGARD TO SIMULATION DEVELOPMENT?

☒ A. High. Extensive experience.
☐ B. Moderate. Some experience, but simulation development is not our core task.
☐ C. Low. This was our first exposure to simulation development.

18. HOW WOULD YOU CHARACTERIZE YOUR LEVEL OF EXPERIENCE WITH REGARD TO THE HLA?

☒ A. High. We have been involved with the program since it's early development.
☐ B. Moderate. We are well versed on the concept and principals, but have never participated in a development effort.
☐ C. Low. Until we got the task, we really hadn't been involved.

19. DID YOU AND OR YOUR DEVELOPMENT TEAM HAVE ACCESS TO OUTSIDE SUPPORT DURING THE TRANSITION?

☒ A. Specialized HLA training sessions (describe):
Some of the development team members have attended regional and hands-on HLA training.

☐ B. HLA Cadre or Mentor Assistance:

☐ C. Commercial Contractor:

20. WERE THERE ANY UNIQUE OBSTACLES THAT HAD TO BE OVERCOME DURING THE TRANSITION PROCESS, IF SO PLEASE DESCRIBE AND PROVIDE AN ESTIMATED LEVEL OF EFFORT IN STAFF TIME?

This effort involved the development and implementation of new HLA object classes and attributes that were not included in the current version of the RPR-FOM. This added an estimated 1-2 man-months worth of effort to implement this within the SOM/FOM and across all applicable federates.

21. CAN YOU CHARACTERIZE THE TECHNICAL APPROACH USED TO ACQUIRE A RUN TIME INFRASTRUCTURE (RTI) INTERFACE FOR YOUR FEDERATE? (Select those that apply)

☒ A. Purchased a commercial product
☐ B. Acquired a Government off the Shelf (GOTS) product or another freeware product
☐ C. Designed and built a middleware application
☐ D. Extended or modified an existing native interface to the Federate
☐ E. Designed and built a native interface to the Federate
☐ F. Used a gateway or other interface device

22. ARE THERE ADDITIONAL ASPECTS OF THE TRANSITION OF YOUR SIMULATION TO THE HLA THAT WARRANT MENTION? IF SO, PLEASE DESCRIBE AND PROVIDE AN ESTIMATED LEVEL OF EFFORT IN STAFF TIME?

The program plan included the development, test, and integration in a test-bed prior to installing the HLA software on-site. This was done to mimimize travel time and on site integration time, and also to minimize simulator down-time to allow for training.

It is also important to mention that this development effort was done in conjunction with several other simulators also undergoing similar HLA migrations. The development and integration time was likely reduced by 1/3 as we took advantage of the synergies among these programs.

The total number of staff months/personnel, as well as the metrics information provided above reflect the effort involved in converting one federate (MC-130H) from DIS to HLA. In reality, the effort was shared among the development of another nearly identical simulator (MC-130E). Thus, the metrics quoted above could have been reduced by 50% to account for this, but this would have been misleading. Instead, the metrics were provided as if there was only one federate.